***Instituto Tecnológico de Costa Rica***

***Unidad de Computación***

***Proyecto Juego Revienta Globos***

***María José Solís García***

***Kevin Varela Rojas***

***Sede San Carlos***

***07/06/2022***

***Introducción***

Hace 72 años con la invención del lenguaje ensamblador, un lenguaje de bajo nivel, brindarles instrucciones a los computadores se volvió un poco más manejable pues “conocer un lenguaje como el Ensamblador y poder, a través de él, construir soluciones a diferentes problemas amplió el abanico de posibilidades de aplicación de los computadores” (Trejos, 2020, pág. 74). Hoy en día con el avance de los lenguajes de programación han aparecido múltiples herramientas que facilitan mucho más la interacción entre un usuario y el ordenador, sin embargo, es importante seguir conociendo sobre los lenguajes de bajo nivel, como ensamblador, ya que estos nos han permitido evolucionar a través de los años y son la base de la programación.

Con el pasar del tiempo, la programación a evolucionado a tal medida que esta no solo se utiliza para girarle ordenes al computador, sino también permite desarrollar aplicaciones que interactúen con los usuarios tal y como lo hacen los videojuegos.

El presente proyecto tiene como objetivo la implementación de un programa en lenguaje ensamblador, construido en la herramienta EMU8086, el cual permita a los usuarios disfrutar del juego sobre reventar globos que aparecen en pantalla, esto con la finalidad de que el jugador se divierta, y al mismo tiempo recordar los lenguajes bases que permitieron el nivel de desarrollo de la programación de la actualidad.

**Análisis del problema**

Al iniciar el juego, se pretende que el usuario observe en pantalla la bienvenida al juego y mediante la tecla “enter” indique que desea continuar, se le pedirá al usuario que introduzca los datos respectivos de cada jugador(nombre y símbolo que representará su globo) Se pretende que el juego cuente con 2 niveles de dificultad en base al tiempo, el primer nivel tiene una duración de alrededor 7 segundos y el nivel dos contará con un aproximado de 3.5 segundos para que el usuario intente explotar la mayor cantidad de globos posibles dando clic sobre ellos. En la pantalla aparecerán ocho globos, los cuales serán representados por el símbolo escogido por cada jugador, estos tendrán que intentar sumar la mayor cantidad de puntos posibles mediante la explosión de los globos, para así ganar el juego. Estos globos aparecerán en distintas posiciones, escogidas mediante coordenadas basadas en números a lazar que se encuentren dentro de las coordenadas de una matriz definida, y serán de diferentes colores. En caso de que el usuario falle al reventar un globo se pretende implementar la reproducción de un “beep”, los globos que sean correctamente capturados se borraran de la matriz y se sumaran a los puntos del jugador respectivo. Cuando el primer jugador termine los dos niveles, continuará el segundo usuario, y luego de que este finalice, se compararán los puntajes y se elegirá el ganador, sin embargo, en ciertos momentos del juego el usuario podrá elegir entre las opciones de rendirse, reiniciar y salir del juego. Si un jugador decide rendirse, su oponente automáticamente se declarará ganador de la partida.

***Solución del problema***

Se define la primera etapa del presente proyecto como la creación de la pantalla de inicio del juego y registro de jugadores, para ello se necesitó construir una pantalla principal, dentro de la directiva data, la cual será la encargada de dibujar la bienvenida al juego en la pantalla. Luego de esto, se construyeron las variables necesarias para el almacenamiento de los datos de nombre de jugador y el símbolo de su globo. Con la finalidad de la implementación y correcto funcionamiento de los datos mencionados anteriormente, dentro de la directiva Code, se creó una macro con el nombre “imprimirPantallas” la cual recibe el dato de una pantalla a imprimir, este macro mueve al registro “ah” la interrupción 09h, encargada de mostrar en pantalla la cadena de caracteres almacenada en memoria, correspondientes al nombre de la pantalla recibida anteriormente, y luego ejecuta la interrupción 21h la cual espera que el usuario pulse una tecla para continuar. Esta macro es ejecutada dentro de la etiqueta llamada “menuPrincipal”, la cual además contiene la llamada a la macro “limpiarPantalla” que se encargaba de mover al registro “ah” la interrupción 0fh, luego se ejecuta la interrupción 10h, encargada de controlar los servicios de pantalla del ordenador, se mueve a “ah” 0, y luego vuelve a ejecutar la interrupción 10h.

Para el registro de nombre de jugadores y símbolos para globos, se definió la etiqueta “usuarios”, esta contiene la llamada a la macro “limpiarPantalla”,y hace alusión también a la macro “imprimirInputs” la cual recibe los valores “texto” y “variable”, referencias para las variables, definidas en la directiva Data, “usuario#”, donde “#” representa el número de jugador, encargada de recibir los datos introducidos por teclado sobre el nombre del jugador y “usuario#\_valor” encargada de guardar los datos de la variable “usuario#”. Del mismo modo, la etiqueta solicita la intervención de la misma macro anteriormente mencionada para el registro de los símbolos de representación de globos, escogido respectivamente por cada jugador, almacenados en la variable “signo#” y posteriormente guardados en la variable “signo#\_valor”.

La etapa número dos de este proyecto será definida como la implementación de la matriz, además del reflejo en pantalla de los datos del jugador y su puntaje. Para el dibujo en pantalla de la matriz se necesitó definir dentro de la directiva Data, las variables “matriz1” y “matriz2”, las cuales contenían las cadenas de caracteres necesarias para dibujar la matriz en la pantalla. En la directiva Code, se creo una etiqueta llamada “nivel1” la cual dentro de ella contenía otra etiqueta llamada “pantallaJuego”, dentro de esta se hace referencia a la macro limpiarPantalla, definida en la etapa anterior, además de esto, se llama a la macro “imprimirTexto”, encargada de recibir una variable a imprimir en pantalla, además de las coordenadas en el eje “x” y “y”, donde se quiere mostrar el texto al usuario. Esta macro mueve al registro “ah” la interrupción 02h , la cual es la encargada de posicionar el cursor en pantalla, luego mueve a “bh” 00d, encargada del numero de página donde se imprimirá el mensaje solicitado, seguidamente mueve a “dh” los datos recibidos de “y” , esto para definir las filas de la matriz, y luego mueve a dl la información del eje “x”, que representa las columnas. A continuación, se ejecuta la interrupción 10h, y luego se mueve al registro “ah” el numero 9, ejecuta la directriz “lea” para el registro “dx” y la variable texto, lo que imprime el texto recibido en pantalla y finalmente ejecuta la interrupción 21h.

Volviendo a la etiqueta “pantallaJuego”, al hacer la llamada a la macro “imprimir texto”, envía los datos de la matriz uno, define su posición en pantalla, y luego vuelve a hacer referencia a esta misma macro, pero enviando los datos de la matriz dos, haciendo referencia a esta un total de 20 veces y finalmente vuelve a hacer referencia a la matriz uno, para así lograr dibujar en pantalla el rectángulo que representa la matriz donde se imprimirán los globos de cada jugador. Además, con la ayuda de la macro anteriormente mencionada, también se imprime las variables de “opciones”, la cual muestra al usuario las oportunidades de renunciar, reinicir y salir durante la ejecución del juego, y también muestra la variable “jugadorActivo”, la cual identifica y señala en pantalla cual jugador realiza su turno en el momento. Ambas variables también se encuentran definidas en la directiva Data.

Dentro de la etiqueta, también se llama a la macro “imprimirDatos”, esta solicita los datos del nombre de usuario y coordenadas donde se quiere imprimir dicho nombre, además de la impresión en pantalla del puntaje de cada jugador, solicitado mediante la misma macro. Dentro de la “pantallaJuego” se encuentra también la etiqueta “estatico”, la cual refiere al “enter” que debe presionar el jugador para comenzar su turno, esta etiqueta mueve a “ah” el servicio 08h, el cual espera a que se pulse una tecla, luego se ejecuta la interrupción 21 y se compara el registro “al” con el código hexadecimal de la tecla “enter”, el cual es 0dh, se realiza un salto condicional a la etiqueta “limpiarTexto” si el registro y 0dh son iguales, sino se realiza un salto a la etiqueta “estatico” nuevamente.

Finalizando la segunda etapa, dentro de “pantallaJuego”, se encuentra la etiqueta “limpiarTexto”, la cual mediante la ayuda de la macro “imprimirTexto”, ejecuta la variable “limpiar”, la cual limpia la frase de solicitud del enter para iniciar el juego, y prepara la matriz para el inicio de la impresión de globos.

Se define la tercera etapa, como la implementación de los globos y niveles del juego para cada jugador. Para esto, dentro de la etiqueta “pantallaJuego”, se crea otra etiqueta llamada “jugador1”, que a su vez contiene otra etiqueta llamada “inicioNivel1J1”, esta define dentro de ella la macro “globo1J1”, la cual recibe el símbolo escogido por el jugador para representar sus globos, seguidamente hace referencia a “numRandom” , una macro que recibe una variable, la cual hace referencia al nombre de la variable definida en la directriz Data como “numRandom”, y luego mueve al registro “ah”, 2ch, encargada de generar el numero al azar, ejecuta la interrupción 21h, y mueve a “variable” el registro “dl” y por último mueve a “al”, “variable”.

En la macro “globo1J1” también se encuentran varias etiquetas que realizan operaciones matemáticas para la correcta aparición de los globos dentro de la matriz del juego; la etiqueta “GY1filaJ1” contiene los procesos definidos como “Y1restaFilaJ1”, encargado de restar 17 unidades al número almacenado en “al” así como “Y1sumaFilaJ1” encargada de sumar al registro “al” 7 unidades; además de esto “GY1filaJ1” contiene la llamada a la macro “Y” la cual recibe las referencias a las etiquetas de operaciones mencionadas anteriormente, “Y” se encarga de comparar el número creado al azar, con diferentes cifras, por ejemplo la comparación con el número 26, la cual realiza un salto condicional a la etiqueta de la resta si el dígito está por encima de este. La comparación realizada con el numero 6 realiza un salto corto a la operación de suma si el número se encuentra por debajo del 6. Por último, en esta macro también se compara con otros números como lo son el 8,10,12,14,16,18,20,22,24,26, todas están comparaciones realizan un salto a la referencia de la etiqueta “Y1filaJ1” si el dígito comparado es igual a los números mencionados anteriormente; esta etiqueta se encarga de almacenar en la variable GX1 el valor del registro “al”, definiendo así el valor de la coordenada del eje x para el primer globo.

Seguidamente se crea otro número random con la ayuda de la macro, pero está vez será dirigido a una nueva etiqueta dentro de “globo1J1”, la cual recibe el nombre de “GX1columnaJ1”, contiene también operaciones de resta y suma para el número generado, con una adición de 7 unidades para el número generado, y una sustracción de 20 unidades, además hace referencia a la macro “X” la cual recibe las referencias a las etiquetas de las operaciones anteriores, y realiza diferentes comparaciones, entre ellas compara el registro “al” con el numero 20, y realiza un salto condicional a la etiqueta respectiva a la resta si el número está por encima de 20; además también realizaba un salto condicional a la operación suma si el número almacenado en “al” se encuentra por debajo o es igual a 1. Si el numero generado es igual a 2,4,6,8,10,12,14,16,18 o 20, se realiza un salto condicional a una variable llamada “columna”, la cual refiere a la etiqueta “X1columnaJ1”, que se encuentra en “GX1columnaJ1”, esta se encarga de mover a la variable GY1, declarada en la directriz Data , el valor correspondiente a la coordenada del eje “y” para el primer globo que se mostrará en pantalla.

Este proceso de creación del globo y almacenamiento de sus coordenadas se retoma para la elaboración de los 7 globos restantes, modificando nada más el nombre de las variables donde se almacenan los datos según el número de globo al que pertenezcan. Luego de determinar estos datos, se solicita la intervención de una macro llamada “coloresGlobos”, la cual recibe el símbolo escogido por el jugador, almacenado en su respectiva variable, y además de un número que representa el valor del color del que se desea que se pinte el globo; esta macro mueve al registro “ah”, la interrupción 9h, mueve a “al” el símbolo almacenado en la variable “signo#\_valor”, asigna al registro “bh” el valor de 0 y al registro “bl” mueve el color solicitado, finalmente al registro “cx” le asigna el valor 1 y ejecuta la interrupción 10h, finalizando la macro.

En el tema de los niveles del juego, se definieron 2, en los cuales su complejidad varía en torno a una cierta cantidad de tiempo posible que posee el jugador para reventar los globos. El primer nivel es delimitado por el nombre “globosNivel1” y hace impresión de los primeros 5 globos definidos anteriormente, además de esto compara un contador, denominado “cont4” en Data, con 2, y realiza un salto condicional a la etiqueta “globosNivel2” si estas variables son iguales, la cual se encargará de llamar a las macros de los globos del nivel 1, más los globos 6,7 y 8, pero, si, por lo contrario, el numero 2 es mayor al registrado en “cont4”, se realizará un salto corto a la etiqueta “clic”. Esta se encarga de comparar la variable “tiempo” con 30, y se realiza un salto condicional, si el tiempo es menor a 30, a la etiqueta “continuar” la cual se encarga de incrementar el tiempo y llamar a la macro “calcularMouse” quien es la encargada de calcular la posición del cursor dentro de la matriz, y compara tanto la captura del clic izquierdo como el del derecho. Por lo contrario, si el tiempo el mayor a 30, se realiza un salto condicional a la etiqueta “cambiarNivel” , la cual llama a la macro “imprimirTexto”, y envía la variable limpiar2, junto con las coordenadas “x” y “y” de cada globo, para así preparar la matriz para la aparición del segundo nivel, también mueve a la variable tiempo, los datos 00h, para inicializarlo nuevamente en cero, incrementa el “cont4” y realiza un salto corto a la etiqueta “jugador1”.

Dentro de “inicioNivel1J1”, también se encuentra una etiqueta llamada “coordenadas”, la cual se encarga de obtener los valores reales de los ejes “x” y “y”, y además existen otras etiquetas llamadas “compararX#J1” donde “#” corresponde al numero del globo, cada una de estas etiquetas compara la posición del clic con la de la coordenada “x” de un globo en específico, si estas son iguales realiza un salto condicional a la etiqueta ”verificar#”, la cual verifica que la coordenada del clic corresponda a la coordenada “y” del globo seleccionado si esto es correcto, se realiza un salto a la etiqueta “compararY#”, la cual incrementa la variable “puntaje1”,definida en Data para almacenar el puntaje del jugador; y además hace el llamado a la macro “imprimirTexto” para que esta se encargue de limpiar el puntaje que se mostraba en pantalla e imprima el nuevo puntaje, además con la ayuda de la macro “limpiarGlobos”, se elimina el globo capturado de la matriz del juego; pero si la coordenada “x” y la del clic no son iguales, se realiza un salto condicional a la etiqueta “compararX#J1”, donde “#” corresponde al numero del siguiente globo registrado. Si definitivamente el las coordenadas del clic capturado no corresponden a ningún globo, se emite un sonido correspondiente a un “beep” indicando que no existe ningún globo en esa posición

Para la partida del jugador dos se realizó, el mismo procedimiento que en la partida del primer jugador, modificando solamente las etiquetas del jugador uno, por las almacenadas para el jugador 2, como lo son su nombre, símbolo para globos y puntaje, en cuanto a los globos las variables utilizadas fueron las mismas que para el jugador 1.

La cuarta etapa fue definida como la implementación de opciones de reinicio, rendirse y salir de la partida de juego, los jugares pueden visualizar estas alternativas en la parte inferior de la pantalla, y acceder a estas dando clic sobre ellas. Al igual que al dar clic sobre un globo, las etiquetas “compararRendirseJ#”, ”compararReiniciarJ#” y “compararSalirJ#”, donde “#” representa el número de jugador, compararan la coordenada x del mouse, con la coordenada x de estas las opciones, si estas resultan ser iguales, compará la coordenada “y” del mouse dentro de las etiquetas “compararYRendirse”, “compararYReiniciar” y “compararYSalir”, respectivamente, si estas son iguales a las almacenadas en memoria, se realizan las siguientes acciones: en el caso de que el jugador solicite rendirse se mostrará la variable que almacena la pantalla que declara al oponente ganador, para la opción de reiniciar, las pantallas serán limpiadas y se volverá a la etiqueta “usuarios” para la solicitud de los datos de los jugadores nuevamente y el inicio de una nueva partida; en el caso de que de la opción para salir del juego, se imprimirá la variable “pantallaFinal4” la cual muestra al usuario el mensaje “Gracias por jugar”.

Finalmente la etapa 5 será delimitada como la etapa de comparación de puntajes y selección del jugador ganador, esto se realiza dentro de la etiqueta “comparacionFinal”, la cual se encuentra dentro de “jugador2”, esta se encarga de mover al registro “cl”, la variable que almacena el puntaje del jugador 1, y la compara con el puntaje del jugador 2, si el primer usuario obtuvo un puntaje mayor al segundo, se realiza el salto condicional a la pantalla que almacena el mensaje para declarar ganador al jugador 1; por lo contrario, si el puntaje es menor, se realiza un salto al mensaje que proclama al segundo usuario ganador; y si los puntajes son iguales, se realiza un salto a la variable “pantFinal3”, la cual muestra que hubo un empate.

***Análisis de resultados***

A continuación, se muestra una tabla que incluye el nombre, estado y observaciones de los resultados obtenidos durante la creación e implementación en ensamblador de las etapas descritas en la solución del problema.

|  |  |  |
| --- | --- | --- |
| ***Tarea*** | ***Estado*** | ***Observaciones*** |
| Creación de la pantalla de inicio del juego y registro de jugadores | *Completado* | En el código implementado, los usuarios pueden registrarse, como se planteó en la solución del problema. Y se creó correctamente la pantalla de inicio del juego. Tal y como se observa en la **figura 1.** |
| Implementación de la matriz, reflejo en pantalla de los datos del jugador y su puntaje. | *Completado* | La matriz fue definida e implementada completamente al igual que el reflejo de los datos del jugador, esto se puede observar en la **figura 2.** |
| Implementación de los globos y niveles del juego para cada jugador | *Completado* | Esta etapa fue completada al máximo tal y como se delimitó en la solución del problema, lo anterior mostrado en la **figura 3.** |
| Implementación de opciones de reinicio, rendirse y salir de la partida de juego | *Completado* | Completo según lo estipulado en la sección de solución del problema, visualizado en la **figura 4.** |
| Comparación de puntajes y selección del jugador ganador | *Completado* | Se puede observar en la **figura 5.** |

**Figura 1:** Creación de la pantalla de inicio del juego y registro de jugadores.

Interfaz de usuario gráfica

Descripción generada automáticamente

Fuente: creación propia

**Figura 2:** Implementación de la matriz, reflejo en pantalla de los datos del jugador y su puntaje.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Fuente: creación propia

**Figura 3:** Implementación de los globos y niveles del juego para cada jugador.

Captura de pantalla de un celular

Descripción generada automáticamente

Fuente: creación propia

**Figura 4:** Implementación de opciones de reinicio, rendirse y salir de la partida de juego.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Fuente: creación propia

**Figura 5:** Comparación de puntajes y selección del jugador ganador.

Interfaz de usuario gráfica

Descripción generada automáticamente

Fuente creación propia

***Conclusiones***

Luego de la implementación del código y el análisis de los resultados, se concluye que:

* El programa cumple las expectativas en cuanto a la implementación de matriz, registro de usuarios y globos, impresión de estos, funcionamiento de los niveles y declaración del ganador, según se esperaba.

***Recomendaciones***

* El programa podría mejorar en cuanto al registro del nombre de usuario, pues actualmente solo registra la inicial de este.
* El juego, en un futuro, puede llegar a contar con más jugadores y niveles, cuya complejidad para avanzar al siguiente se base en la cantidad de globos atrapados y no en un temporizador.

***Referencias***

DIRTYCODE11, D. (2021). *CODIGOS-EN-GENERAL/mousee.asm at main · DIRTYCODE11/CODIGOS-EN-GENERAL*.GitHub. https://github.com/DIRTYCODE11/CODIGOS-EN-GENERAL/blob/main/assembler%20(TASM)/mousee.asm

*Ensamblador - Posicionar cursor*. (2010). YouTube. https://www.youtube.com/watch?v=KALqmJSysVU&ab\_channel=facomsys

Facomsys. (2010). *Ensamblador - Posicionar cursor* [Vídeo]. YouTube. https://www.youtube.com/watch?v=KALqmJSysVU&ab\_channel=facomsys

MEYER PROGRAMA. (2020). *Vídeo #7: Imprimir caracteres a color* [Vídeo]. YouTube. https://www.youtube.com/watch?v=CieoLyvXPMg&ab\_channel=MEYERPROGRAMA

MEYER PROGRAMA. (2021). *Clase 7. ¿Cómo imprimir un texto en cualquier lugar de la pantalla?* [Vídeo]. YouTube. https://www.youtube.com/watch?v=gFC44Jd8kgA&ab\_channel=MEYERPROGRAMA

Sedory, D. (2017). *Documentation for emu8086 - assembler and microprocessor emulator*. emu8086. file:///C:/emu8086/documentation/index.html